

1 **In the Claims**

2 Claims 1-78 remain in the application and are listed as follows:

3
4 **CLAIMS**

5 1. (Original) A method for testing software comprising:
6 modeling software using a software model that describes behavior
7 associated with the software; and
8 operating on the software model using a random destination algorithm and
9 at least one other different algorithm to produce a sequence of test actions, the
10 random destination algorithm being configured to randomly select a destination in
11 the model and move to that destination to produce the sequence of test actions.

12
13 2. (Original) The method of claim 1, wherein the software model
14 comprises state graph having multiple nodes individual ones of which representing
15 a state, and links between the nodes that represent actions.

16
17 3. (Original) The method of claim 2, wherein said operating comprises
18 using the random destination algorithm to select a destination node at random,
19 independent of a present node, and traverse state space to arrive at the destination
20 node.

1 4. (Original) The method of claim 2, wherein said operating comprises
2 using the random destination algorithm to select a destination node at random,
3 independent of any previously-traversed nodes, and traverse state space to arrive at
4 the destination node.

5
6 5. (Original) The method of claim 2, wherein said operating comprises
7 using the random destination algorithm to select a destination node at random,
8 independent of a nearest neighbor node, and traverse state space to arrive at the
9 destination node.

10
11 6. (Original) The method of claim 2, wherein the software model
12 comprises clusters of related nodes, and said operating comprises using the
13 random destination algorithm to select, at random, at least one cluster of nodes.

14
15 7. (Original) The method of claim 2, wherein the software model
16 comprises clusters of related nodes, and said operating comprises using the
17 random destination algorithm to select, at random, at least one node inside at least
18 one cluster of nodes.

19
20 8. (Original) One or more computer-readable media having computer-
21 readable instructions thereon which, when executed by a computer, cause the
22 computer to:

23 model software using a software model that describes behavior associated
24 with the software, the software model comprising a state graph having multiple
25

1 nodes individual ones of which represent a state, and links between the nodes that
2 represent actions; and

3 operate on the software model using a random destination algorithm and at
4 least one other different algorithm to produce a sequence of test actions, the
5 random destination algorithm being configured to randomly select a destination
6 node in the model and move to that destination node to produce the sequence of
7 test actions, the selection of the destination node being performed independent of
8 any previously-traversed nodes, and independent of any nearest neighbor nodes.

9
10 9. (Original) A method of testing software comprising:

11 modeling software using a software model that describes behavior
12 associated with the software;

13 operating on the software model using a random destination algorithm to
14 produce a sequence of test actions, the random destination algorithm being
15 configured to randomly select a destination in the model and move to that
16 destination to produce the sequence of test actions; and

17 operating on the software model using multiple other algorithms that are
18 different from the random destination algorithm to produce a further sequence of
19 test actions.

20
21 10. (Original) The method of claim 9, wherein said modeling comprises
22 using a state graph having multiple nodes individual ones of which represent a
23 state, and links between the nodes that represent actions.

1 11. (Original) The method of claim 9, wherein said multiple other
2 algorithms comprise a random walk algorithm.

3
4 12. (Original) The method of claim 9, wherein said multiple other
5 algorithms comprise a Chinese postman algorithm.

6
7 13. (Original) The method of claim 9, wherein said multiple other
8 algorithms comprise a Markov chain algorithm.

9
10 14. (Original) The method of claim 9, wherein said multiple other
11 algorithms comprise a anti-random walk algorithm.

12
13 15. (Original) The method of claim 9, wherein said multiple other
14 algorithms comprise an algorithm selected from a group comprising: a random
15 walk algorithm, a Chinese postman algorithm, a Markov chain algorithm, and a
16 anti-random walk algorithm.

17
18 16. (Original) One or more computer-readable media having computer-
19 readable instructions thereon which, when executed by a computer, cause the
20 computer to:

21 operate on a software model using a random destination algorithm to
22 produce a sequence of test actions, the software model comprising a state graph
23 having multiple nodes individual ones of which represent a state, and links
24 between the nodes that represent actions, the random destination algorithm being
25

1 configured to randomly select a destination node in the state graph and move to
2 that destination node to produce the sequence of test actions; and

3 operate on the software model using multiple other algorithms that are
4 different from the random destination algorithm to produce a further sequence of
5 test actions, the multiple other algorithms being selected from a group comprising:
6 a random walk algorithm, a Chinese postman algorithm, a Markov chain
7 algorithm, and a anti-random walk algorithm.

8
9 17. (Original) A method of testing software comprising:
10 traversing a state graph that models software, the state graph having
11 multiple nodes individual ones of which represent a state, and links between the
12 nodes that represent actions, said traversing using an algorithm having a first
13 graph traversal characteristic to produce a sequence of test actions; and

14 traversing the state graph using an algorithm having a second graph
15 traversal characteristic that is different from the first graph traversal characteristic
16 to produce a further sequence of test actions.

17
18 18. (Original) The method of claim 17, wherein the algorithms are
19 different.

20
21 19. (Original) The method of claim 17, wherein the algorithm having the
22 first graph traversal characteristic is one selected from a group of algorithms
23 comprising: a random walk algorithm, a random destination algorithm, and a anti-
24 random walk algorithm.

1 20. (Original) The method of claim 19, wherein the algorithm having the
2 second graph traversal characteristic different from the algorithm having the first
3 graph traversal characteristic and is one selected from a group of algorithms
4 comprising: a random walk algorithm, a random destination algorithm, and a anti-
5 random walk algorithm.

6
7 21. (Original) One or more computer-readable media having computer-
8 readable instructions thereon which, when executed by a computer, cause the
9 computer to implement the method of claim 17.

10
11 22. (Original) A method of testing software comprising:
12 traversing a state graph using a deterministic first algorithm to produce a
13 sequence of test actions, the state graph having multiple nodes individual ones of
14 which represent a state, and links between the nodes that represent actions; and
15 traversing the state graph using a second algorithm that is less deterministic
16 than the first algorithm to produce a further sequence of test actions.

17
18 23. (Original) A method of testing software comprising:
19 traversing a state graph using a random walk first algorithm to produce a
20 sequence of test actions, the state graph having multiple nodes individual ones of
21 which represent a state, and links between the nodes that represent actions; and
22 traversing the state graph using a second algorithm that is less random than
23 the first algorithm to produce a further sequence of test actions.

1 24. (Original) A method of testing software comprising:
2 providing one or more algorithms for operating on a software model that
3 describes behavior associated with software that is to be tested;
4 selecting one or more algorithms;
5 operating on the software model using the selected one or more algorithms
6 to produce a sequence of test actions;
7 changing the selected one or more algorithms; and
8 operating on the software model using one or more changed algorithms.

9
10 25. (Original) The method of claim 24, wherein said changing comprises
11 changing a way an algorithm interacts with the software model.

12
13 26. (Original) The method of claim 25, wherein said changing comprises
14 changing one or more properties associated with an algorithm.

15
16 27. (Original) The method of claim 24, wherein said changing comprises
17 selecting at least one different algorithm.

18
19 28. (Original) One or more computer-readable media having computer-
20 readable instructions thereon which, when executed by a computer, cause the
21 computer to:

22 provide one or more algorithms for operating on a software model that
23 describes behavior associated with software that is to be tested;
24 select multiple algorithms to define a first collection of algorithms;
25

1 operate on the software model using the first collection of algorithms to
2 produce a sequence of test actions;

3 change at least one of the selected algorithms to define a second collection
4 of algorithms; and

5 operate on the software model using the second collection of algorithms to
6 produce an additional sequence of test actions.

7
8 29. (Original) A method of testing software comprising:

9 traversing a state graph using a random destination algorithm, the state
10 graph having multiple nodes individual ones of which representing a state, and
11 links between the nodes that represent actions, said traversing producing a
12 sequence of test actions; and

13 traversing the state graph using multiple steps from a random walk
14 algorithm to produce an additional sequence of test actions.

15
16 30. (Original) The method of claim 29 further comprising traversing the
17 state graph using a random destination algorithm after said traversing of the state
18 graph using the random walk algorithm.

19
20 31. (Original) The method of claim 29, wherein said traversing using
21 multiple steps comprises using a predetermined number of steps.

22
23 32. (Original) The method of claim 29, wherein said traversing using
24 multiple steps comprises using a random number of steps.
25

1 33. (Original) The method of claim 29, wherein said acts of traversing
2 comprise iterating through the random destination and random walk algorithms.

3
4 34. (Original) The method of claim 33, wherein said traversing using
5 multiple steps comprises changing the number of steps on at least one iteration.

6
7 35. (Original) The method of claim 33, wherein said traversing using
8 multiple steps comprises randomly changing the number of steps on at least one
9 iteration.

10
11 36. (Original) The method of claim 33, wherein said traversing using
12 multiple steps comprises changing the number of steps on at least one iteration in
13 accordance with probabilistic characteristics.

14
15 37. (Original) A method of testing software comprising:
16 selecting a first algorithm from among a number of different algorithms;
17 operating on a software model that describes behavior of software that is to
18 be tested, said operating taking N steps using the first algorithm, where N is an
19 integer and said steps produce a sequence of test actions;
20 selecting a second algorithm from among the number of different
21 algorithms, the second algorithm being different from the first algorithm; and
22 operating on the software model by taking N1 steps using the second
23 algorithm, where N1 is an integer, said N1 steps producing an additional sequence
24 of test actions.
25

1 38. (Original) The method of claim 37, wherein the algorithms are
2 different based upon how they interact with the software model.

3
4 39. (Original) The method of claim 37, wherein the algorithms are
5 different based upon how they traverse a graph that describes the software's
6 behavior.

7
8 40. (Original) The method of claim 37, wherein said number of different
9 algorithms include at least one algorithm that is more deterministic than at least
10 one other algorithm.

11
12 41. (Original) The method of claim 37, wherein said number of different
13 algorithms include at least one algorithm that is more random than at least one
14 other algorithm.

15
16 42. (Original) The method of claim 37, wherein at least one of said acts
17 of selecting is based on the structure of the software model.

18
19 43. (Original) The method of claim 37, wherein N and N1 are
20 predetermined.

21
22 44. (Original) The method of claim 37, wherein N and N1 are
23 preprogrammed.

1 45. (Original) The method of claim 37, wherein N and N1 are randomly
2 selected.

3
4 46. (Original) The method of claim 37, wherein N and N1 are calculated
5 using a Poisson distribution having multiple values each with an assigned
6 probability of being selected.

7
8 47. (Original) The method of claim 46, wherein one or more assigned
9 probabilities change over time.

10
11 48. (Original) The method of claim 37 further comprising iterating
12 through said acts of operating at least one time.

13
14 49. (Original) The method of claim 48 further comprising changing the
15 values of N and N1 during the iteration.

16
17 50. (Original) The method of claim 48 further comprising:
18 assigning values to N and N1 using a first method on a first pass; and
19 assigning values to N and N1 using a second method that is different from
20 the first method on a second pass.

21
22 51. (Original) The method of claim 37 further comprising replacing one
23 or more of the algorithms used to operate on the software model after a certain
24 period of time.

1 52. (Original) The method of claim 37 further comprising replacing one
2 or more of the algorithms used to operate on the software model after the one or
3 more algorithms have been used a certain number of times.

4
5 53. (Original) A method of testing software comprising:
6 representing software using a model that describes the software's behavior,
7 the software having an associated social context; and
8 selecting one or more algorithms to operate upon the model as a function of
9 the software's social context; and
10 operating upon the model using the selected one or more algorithms to
11 produce a sequence of test actions.

12
13 54. (Original) The method of claim 53, wherein the social context is
14 associated with a software developer who developed the software.

15
16 55. (Original) The method of claim 53 further comprising:
17 changing the one or more algorithms; and
18 operating upon the model using changed algorithms to produce an
19 additional sequence of test actions.

20
21 56. (Original) A method of testing software comprising:
22 developing a profile associated with one or more software developers, the
23 profile describing one or more algorithms that are more likely to identify problems
24 associated with software developed by the one or more software developers;
25

1 selecting, from a developer's profile, one or more algorithms when a
2 software model associated with the developer's software is to be operated upon;
3 and

4 operating upon the software model using the selected one or more
5 algorithms to produce a sequence of test actions.

6
7 57. (Original) A method of testing software comprising:

8 defining one or more clusters in a software model that models software that
9 is to be tested;

10 providing multiple different algorithms for operating upon the software
11 model;

12 selecting a first algorithm for operating on the software model to produce a
13 sequence of test actions;

14 selecting a second algorithm that is different from the first algorithm for
15 operating on the software model to produce an additional sequence of test actions;
16 and

17 operating on the software model using the first and second algorithms to
18 produce the sequences of test actions, one of the first and second algorithms
19 having a better chance at accessing a cluster than the other of the first and second
20 algorithms.

1 58. (Original) The method of claim 57, wherein said software model
2 comprises a state graph having multiple nodes and links between the nodes,
3 individual nodes representing states, individual links representing actions that
4 move between states.

5
6 59. (Original) The method of claim 58, wherein the first and second
7 algorithms have different graph traversal characteristics.

8
9 60. (Original) The method of claim 58, wherein said defining comprises
10 defining the clusters based on areas of connectivity within the state graph.

11
12 61. (Original) The method of claim 57, wherein said defining comprises
13 defining the clusters based on the structure of the software.

14
15 62. (Original) The method of claim 57, wherein one of said first and
16 second algorithms comprises a random destination algorithm.

17
18 63. (Original) The method of claim 57 wherein one of said first and
19 second algorithms comprises a random walk algorithm.

20
21 64. (Original) The method of claim 57, wherein one of said first and
22 second algorithms comprises a anti-random walk algorithm.

1 65. (Original) The method of claim 57, wherein one of said first and
2 second algorithms comprises a deterministic algorithm, and the other algorithm
3 comprises a non-deterministic algorithm.

4
5 66. (Original) A software-testing system comprising:

6 a software model processor configured to:

7 receive a software model that describes behavior associated with
8 software that is to be tested, and

9 operate upon the model to provide a sequence of test commands for
10 testing the software; and

11 an algorithm set associated with the model processor and comprising
12 multiple different algorithms, the software model processor being configured to
13 select at least two different algorithms and use the algorithms to operate upon the
14 software model to produce the sequence of test commands.

15
16 67. (Original) The software-testing system of claim 66, wherein the
17 model processor is configured to change one or more of the algorithms.

18
19 68. (Original) The software-testing system of claim 66, wherein at least
20 one algorithm comprises a random walk algorithm.

21
22 69. (Original) The software-testing system of claim 66, wherein at least
23 one algorithm comprises a Chinese Postman algorithm.

1 70. (Original) The software-testing system of claim 66, wherein at least
2 one algorithm comprises a Markov chain algorithm.

3
4 71. (Original) The software-testing system of claim 66, wherein at least
5 one algorithm comprises a anti-random walk algorithm.

6
7 72. The software-testing system of claim 66, wherein at least one
8 algorithm is selected from a group comprising a random walk algorithm, a
9 Chinese Postman algorithm, a Markov chain algorithm, and a anti-random walk
10 algorithm.

11
12 73 (Original) A software-testing system comprising:
13 a software model processor configured to:
14 receive a software model in the form of a state graph that describes
15 behavior associated with software, the state graph having multiple nodes
16 that represent state, and links between the nodes that represent actions, and
17 traverse the state graph to provide a sequence of commands for
18 testing the software;
19 an algorithm set associated with the model processor and comprising
20 multiple different algorithms; and
21 a graph traverser associated with the model processor and configured to:
22 traverse the state graph using an algorithm from the algorithm set,
23 the algorithm having a first graph traversal characteristic to produce a
24 sequence of test commands, and
25

1 traverse graph with an algorithm from the algorithm set having a
2 second graph traversal characteristic that is different from the first graph
3 traversal characteristic to produce a further sequence of test commands.
4

5 74. (Original) The software-testing system of claim 73, wherein the
6 algorithm having the first graph traversal characteristic is selected from a group of
7 algorithms comprising: random walk algorithms, random destination algorithms,
8 and anti-random walk algorithms.
9

10 75. (Original) The software-testing system of claim 74, wherein the
11 algorithm having the second graph traversal characteristic is selected from a group
12 of algorithms comprising: random walk algorithms, random destination
13 algorithms, and anti-random walk algorithms.
14

15 76. (Original) A software-testing system comprising:
16 means for receiving a software model;
17 means for operating on the software model in a first manner to produce a
18 sequence of test actions; and
19 means for operating on the software model in different additional manners
20 to produce additional sequences of test actions.
21

22 77. (Original) The software-testing system of claim 76, wherein said
23 means for operating comprises multiple different graph traversal algorithms.
24
25

1 78. (Original) A method of modeling user behavior comprising:
2 representing software using a model comprising a state graph, the state
3 graph having multiple nodes individual ones of which represent a state, and links
4 between the nodes that represent actions;
5 traversing the state graph using an algorithm having a first graph traversal
6 characteristic to produce a sequence of user actions; and
7 traversing the state graph using an algorithm having a second graph
8 traversal characteristic that is different from the first graph traversal characteristic
9 to produce a further sequence of user actions.
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25